

REMARKS/ARGUMENTS

Claims 1-31 and 42-48 are pending. New claims 42-48 are added and claims 32-41 are canceled. Claims 1, 3, 6, 14, 16-19, 27, 28, and 31 are amended.

Claims 1- 41 are rejected under 35 U.S.C. 102(e) as being anticipated by Apuzzo et al. (US 6,986,125). Applicant submits that all of the claims currently pending in this application are patentably distinguishable over the cited references for the following reasons, and reconsideration and allowance of this application are respectfully requested.

Independent claims 1 and 14 include, among other elements, "wherein each of the plurality of software verification tools corresponds to a respective lifecycle phase of the computer software, and automatically generates one or more test cases from the source code of the computer software," "generating verification results for each respective lifecycle phase of the computer software, responsive to executing the plurality of software verification tools and the automatically generated test cases," and "processing the verification results for generating a representation of functional behavior of the computer software." Apuzzo does not teach the above elements.

Rather, the system of Apuzzo performs functional by using "an abstraction matrix" that describes a software component to be tested. The abstraction matrix includes at least one test case scenario and mapped expected results therefore, that is, an already formulated and known expected test results. The test results are automatically evaluated using the abstraction matrix by comparing the test result for an executed test case with the mapped expected result in the abstraction matrix. (See abstract, FIG. 13, and related text).

First, regarding the element of "each of the plurality of software verification tools corresponds to a respective lifecycle phase of the computer software," Apuzzo does not disclose this element. Applicant respectfully disagrees with Examiner characterization of the "List of Abstractions" of FIGs. 3 and 4 of Apuzzo, as the claimed "plurality of software verification tools." As described above, the Abstractions of Apuzzo are merely "test case scenario and mapped expected results," and not software verification tools, as known by one with ordinary

skill in the art of computer software programming and as defined in the specification. More importantly, even if, for the sake of argument, the Abstractions of Apuzzo can be construed to be software verification tools, they do not correspond "to a respective lifecycle phase of the computer software."

Second, with respect to the element of each of the plurality of software verification tools "automatically generates one or more test cases from the source code of the computer software," Apuzzo does not teach this element either. Again, if for the sake of argument, the Abstractions of Apuzzo can be construed to be software verification tools, they do not automatically generate test cases from the source code of the computer software." Apuzzo is clear about how its test cases are generated: "an abstraction 340 of the software specification is created which is then employed to create individual test cases for the different layers of the software component. The abstraction . . . comprises in one embodiment a matrix that describes the software component." (Col. 5, lines 38-43). Apuzzo further stresses that "the mapped expected results comprise specific states of the software component and test cases are generated associated with the particular mapped expected results." Therefore, it is clear that the test cases of Apuzzo are NOT created from the source code of the computer software, rather, they are created from an already existing Abstraction matrix that needs to be developed, before the test cases can be generated. In contrast, the claimed invention generates the test cases from the source code itself, with the need to develop an Abstraction Matrix. This eliminates the substantial and tedious step of developing the Abstraction Matrix for each software to be tested.

Third, regarding the element of "generating verification results for each respective lifecycle phase of the computer software," Apuzzo does not teach this, because, as explained above, Apuzzo's Abstraction does not correspond to respective lifecycle phases of the computer software.

Fourth, with respect to the element of "processing the verification results for generating a representation of functional behavior of the computer software." Apuzzo does not teach the above element. Rather, as described above, Apuzzo evaluates the test results using the abstraction matrix by comparing the test result for an executed test case with the mapped

expected result in the abstraction matrix. (See abstract, FIG. 13, and related text, underlining added.).

Consequently, for at least each of the above four reasons, Apuzzo does not teach all elements of claims 1 and 14 and therefore, claims 1 and 14 are not anticipated by Apuzzo.

Independent claims 28 and 45 include, among other elements, "providing a known error in the computer software, the known error belonging to a class of errors," "providing a plurality of software verification tools each of the plurality of software verification tools corresponding to a respective lifecycle phase of the computer software," "analyzing the known error in the computer software to determine what phase of the lifecycle the error was introduced," and "customizing a verification scope of one or more of the plurality of verification tools that correspond to the lifecycle phase that the known error was introduced." Apuzzo does not teach the above elements.

First, regarding the element of "providing a known error in the computer software, the known error belonging to a class of errors," there is no teaching of providing a known error in Apuzzo. Rather, Apuzzo's objective is to perform a functional test by comparing test results to expected results in the Abstraction Matrix. There is no known error in Apuzzo being provided to the code.

Second, the element of "providing a plurality of software verification tools each of the plurality of software verification tools corresponding to a respective lifecycle phase of the computer software," as explained above, is not disclosed by Apuzzo.

Third, the element of "analyzing the known error in the computer software to determine what phase of the lifecycle the error was introduced," is also not disclosed by Apuzzo. As discussed above, with respect to claims 1 and 14, Apuzzo's Abstraction and verification process do not correspond to respective lifecycle phases of the computer software.

Fourth, with respect to "customizing a verification scope of one or more of the plurality of verification tools that correspond to the lifecycle phase that the known error was introduced," Apuzzo does not teach i) plurality of verification tools, ii) verification scope for each of the

Appln No. 10/613,166
Amdt date February 6, 2007
Reply to Office action of December 15, 2006

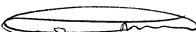
verification tools, iii) customization of the verification scope, and finally iv) determining what phase of the lifecycle the error was introduced.

As a result, for at least each of the above four reasons, Apuzzo does not teach all elements of claims 28 and 45 and thus, claims 28 and 45 are not anticipated by Apuzzo either.

Dependent claims 2-13, 15-26, 28-31, 42-44, and 46-48 are dependent from allowable independent claims 1, 14, 27, and 45, respectively and therefore include all the limitations of their base claims and additional limitations therein. Accordingly, these claims are also allowable over the cited references, as being dependent from an allowable independent claim and for the additional limitations they include therein.

In view of the foregoing amendments and remarks, it is respectfully submitted that this application is now in condition for allowance, and accordingly, reconsideration and allowance are respectfully requested.

Respectfully submitted,
CHRISTIE, PARKER & HALE, LLP

By 

Raymond R. Tabandeh
Reg. No. 43,945
626/795-9900

RRT/clv

CLV PAS722391.1-*02/6/07 2:42 PM